PASSED

Computer Lab 6

Andrea Bruzzone, Thomas Zhang

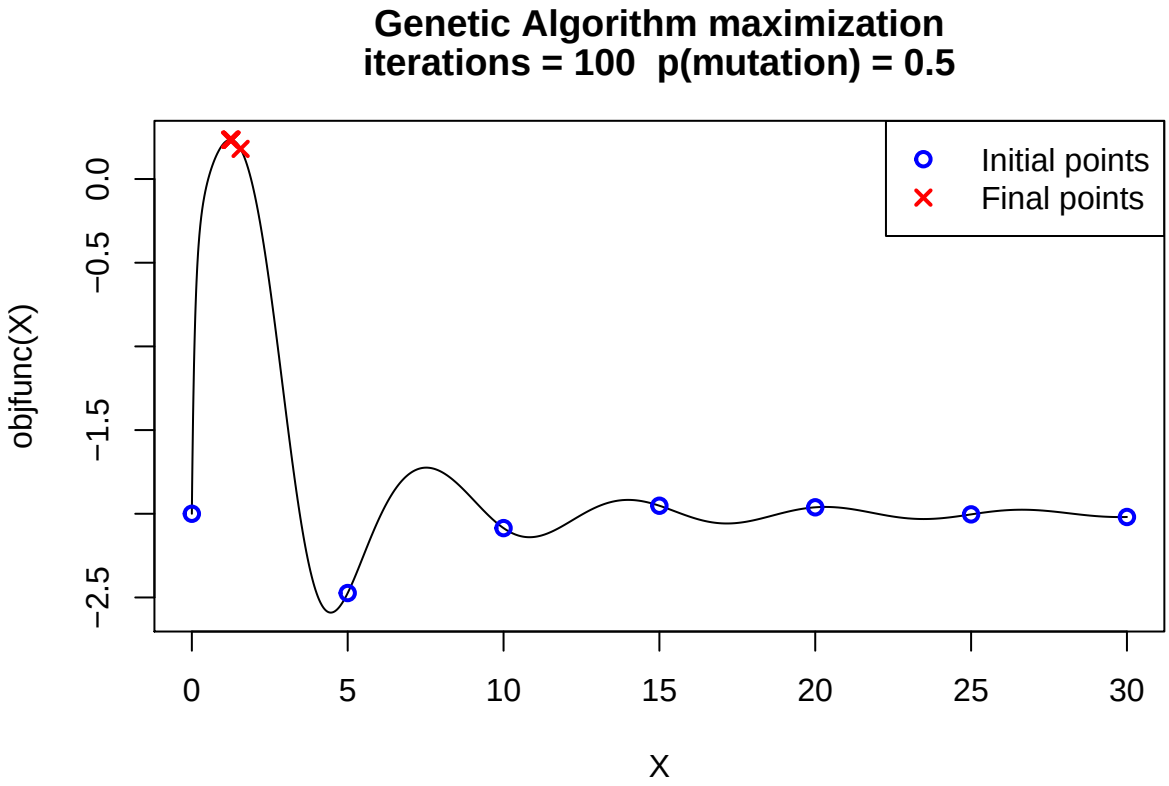
2016 M03 10

Assignment 1

In this assignment we work with the function:

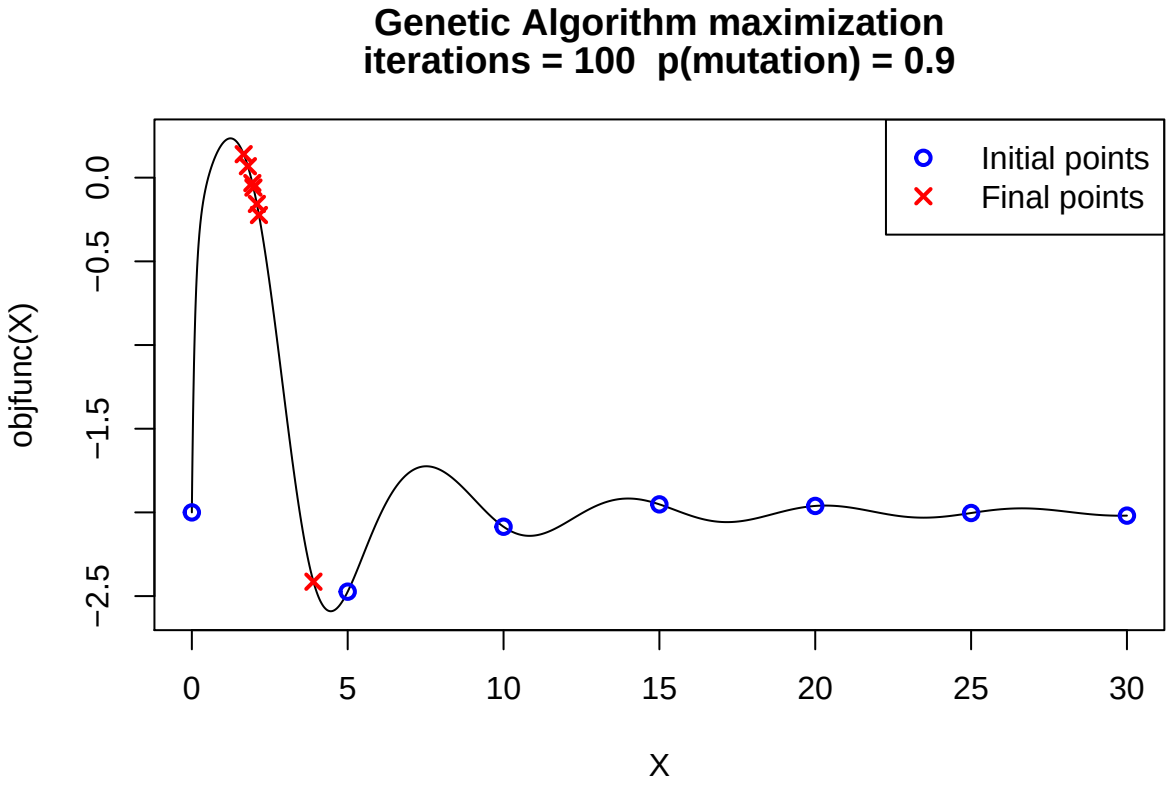
$$f(x) = \frac{x^2}{e^x} - 2e^{\frac{-9sin(x)}{x^2+x+1}}$$

and we try to perform one-dimensional maximization using the genetic algorithm specified in the instructions. The functions can be found in the code part of this report.



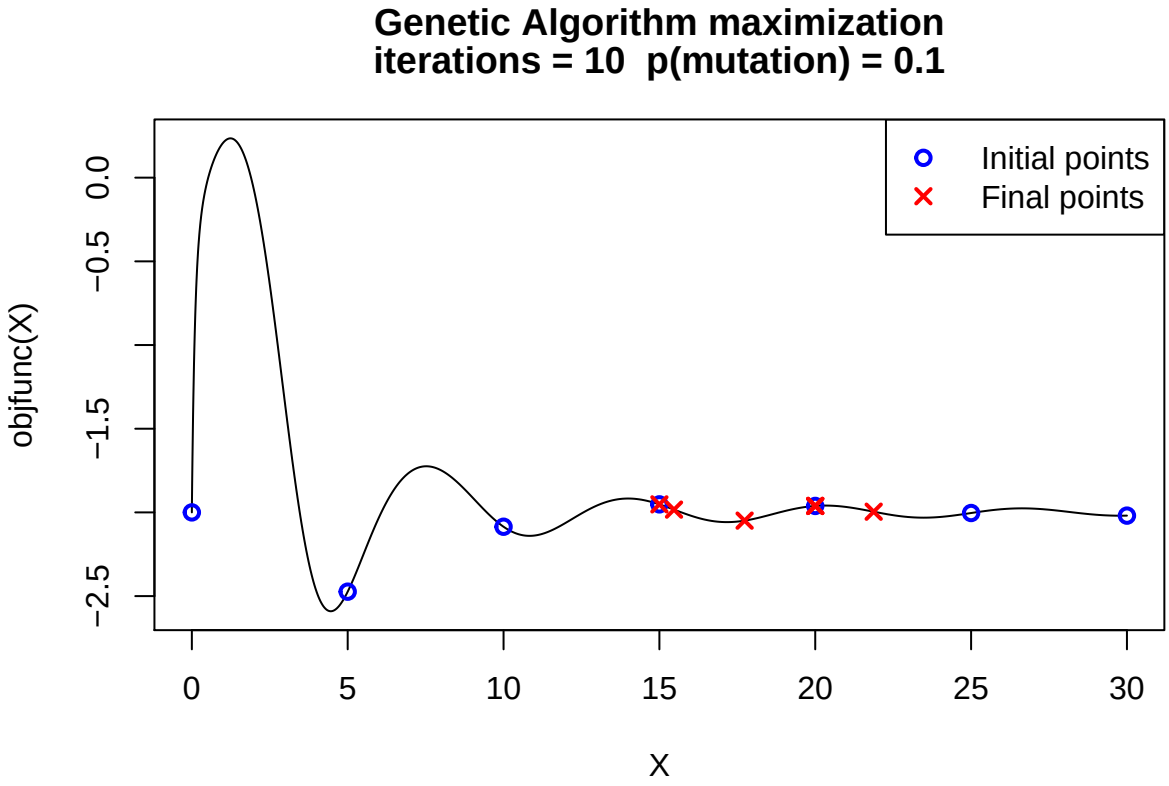

```
## [1] "Final maximum value achieved: 0.2348"
```

As we can see, setting `maxiter = 100` and `p(mutation) = 0.5` gives rather good results, in the sense that the set of final points all are located close to the true maximum. The setting `maxiter = 100` and `p(mutation) = 0.9` produces similar results but the final points are more spread out along the function curve, due to the higher mutation rate.




```
## [1] "Final maximum value achieved: 0.1403"
```

To contrast, we can try setting `maxiter = 10` and `p(mutation) = 0.1`, which often results in the final points being located not far from where they started.




```
## [1] "Final maximum value achieved: -1.9519"
```

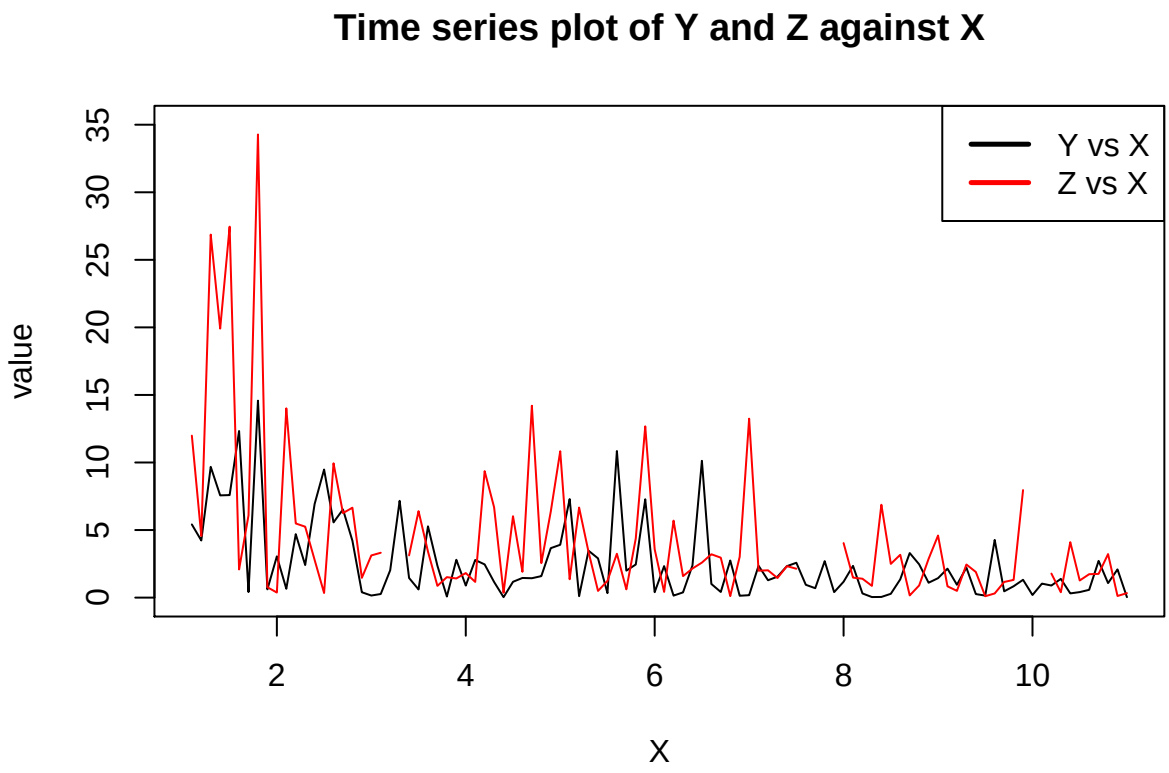
We can infer that the genetic algorithm used requires more than 10 iterations to be effective and that the mutation rate should be moderately high for best results.

Assignment 2

The dataset `physical.csv` describes a behavior of two related physical process Y=Y(X) and Z=Z(X).

- 2.1

The plot shows together Z vs X and Y vs X. Notice the missing values of Z around $X = 3, 8, 10$.



It seems that the two process are related, the Z process seems to be the Y process amplified and translated to the right. Moreover, it can be seen that the highest values for both the process we have for X between 0 and 2, then the two process decrease in values but they continue to be quite random.

- 2.2

It can be seen that Z has some missing values, so we use the EM algorithm to estimate λ.

We compute the likelihood for $Y_i \sim Exp(\frac{X_i}{\lambda})$ and $Z_i \sim Exp(\frac{X_i}{2\lambda})$:

$$L(Y|\lambda) = \frac{\prod_{i=1}^{n} X_i}{\lambda^n} \exp\left(-\frac{\sum_{i=1}^{n} Y_i X_i}{\lambda}\right)$$

$$L(Z|\lambda) = \frac{\prod_{i=1}^{n} X_i}{2^n \lambda^n} \exp\left(-\frac{\sum_{i=1}^{n} Z_i X_i}{2\lambda}\right) = \frac{\prod_{i=1}^{n} X_i}{2^n \lambda^n} \exp\left(-\frac{\sum_{O} Z_i X_i}{2\lambda} - \frac{\sum_{M} Z_i X_i}{2\lambda}\right)$$

where O is the set of indices for the observed values of Z and M is the set of indices for the missing values of Z.

Then, we take the logarithm to get:

$$l(Y, Z|\lambda) = \log \frac{(\prod_{i=1}^{n} X_i)^2}{2^n \lambda^{2n}} - \frac{\sum_{i=1}^{n} Y_i X_i}{\lambda} - \frac{\sum_O Z_i X_i}{2\lambda} - \frac{\sum_M Z_i X_i}{2\lambda}$$

At this point the E-step can be done. We are going to set every missing Z_i to its Expected value given X_i and the last lambda value, λ_t.

$$E[Z_i|X_i, \lambda_t] = \frac{2\lambda_t}{X_i}$$

Due to the exponential distribution of Z_i.

$$E(l(Y, Z|\lambda)) = \log \frac{(\prod_{i=1}^{n} X_i)^2}{2^n \lambda^{2n}} - \frac{\sum_{i=1}^{n} Y_i X_i}{\lambda} - \frac{\sum_O Z_i X_i}{2\lambda} - \frac{|M|\lambda_t}{\lambda}$$

Where $|M|$ is the number of missing Z values. For the M-step, we have to compute the derivative with respect to λ and put it equal to zero.

Doing this we get:

$$\lambda_{t+1} = \frac{\sum_{i=1}^{n} Y_i X_i}{2n} + \frac{\sum_O Z_i X_i}{4n} + \frac{|M|\lambda_t}{2n}$$

- 2.3

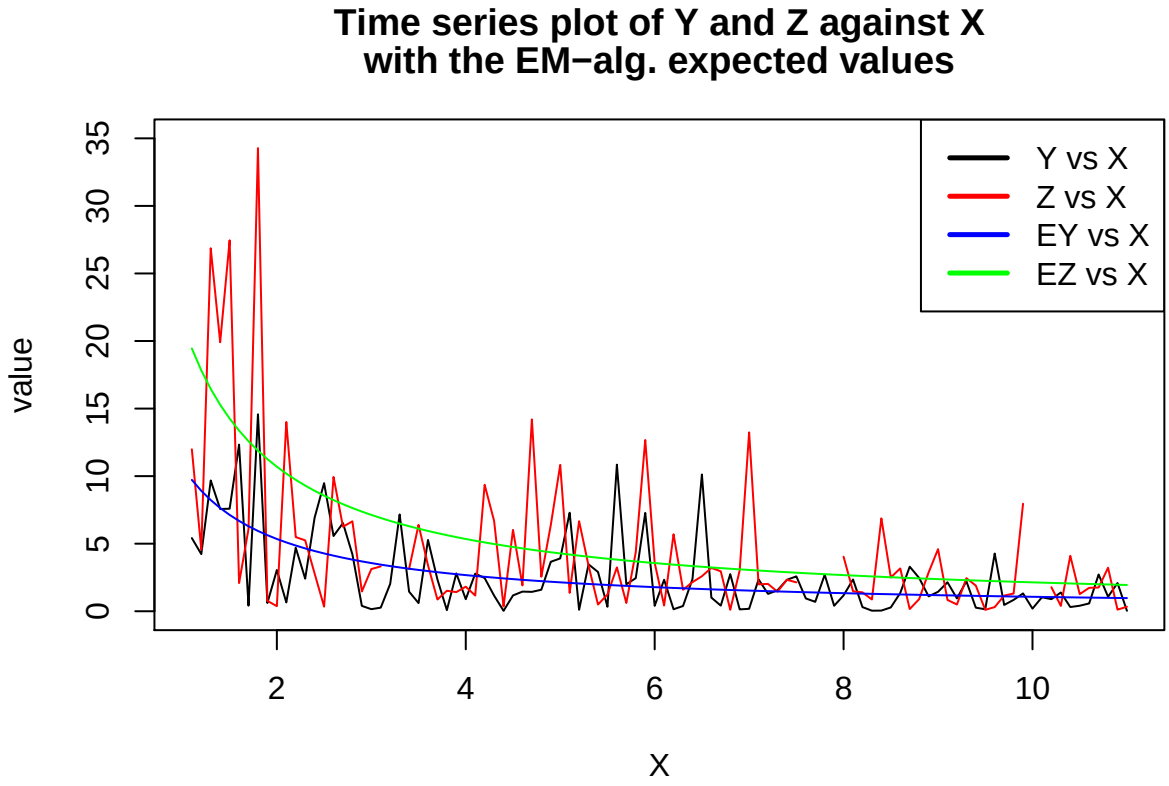
We implement the algorithm in R using a starting λ value of $\lambda_0 = 100$ and converge criterion: stop if the change in λ is less than 0.001.

The optimal value and the number of iterations are:

```
##   Iterations   Lambda
## 1          5 10.69587
```

- 2.4

Using the optimal value of λ found in the previous step, we compute the mean for Y and Z and we plot it against X in the same plot as in assignment 2.1.



The computed λ seems to be reasonable, based on visual inspection of the fit to data.

Group Contributions

Andrea did heroic work with the latex for the report, while Thomas contributed the nice code for the genetic algorithm plots. We discussed the EM-derivation thoroughly and managed to correct each others mistakes. First assignment is derived from Thomas code while second assignment comes from Andreas code.

Appendix

R code

```
objfunc <- function(x){
  res <- x^2/exp(x) - 2 * exp(-9 * sin(x) / (x^2 + x + 1))
  return(res)
}
crossover <- function(x,y){
  return((x + y) / 2)
}
mutate <- function(x){
  res <- x^2 %% 30
  return(res)
}
```

```
genalgfunc <- function(maxiter = 100 , mutprob = 0.5){
  X <- seq(0,30,0.01)
  plot(X,objfunc(X), main = c("Genetic Algorithm maximization",
                              paste("iterations =",maxiter,
                                    " p(mutation) =",mutprob)),
       type="l")
  X <- seq(0,30,5)
  Values <- objfunc(X)
  points(X,Values,col = "blue",cex = 1, pch = 1,lwd = 2)
  count <- 0
  maxvals <- c()
  repeat{
    if(count == maxiter){
      break
    }
    parents <- sample(X,2)
    victim <- order(Values)[1]
    kid <- crossover(parents[1],parents[2])
    if(runif(1) < mutprob){
      kid <- mutate(kid)
    }
    X[victim] <- kid
    Values <- objfunc(X)
    maxvals <- c(maxvals,max(Values))
    count <- count + 1
  }
  points(X,Values,col = "red",cex = 1, pch = 4,lwd = 2)
  legend("topright",legend = c("Initial points","Final points"),
         lty = c(0,0), col= c("blue","red"), pch = c(1,4), lwd = c(2,2))
  paste("Final maximum value achieved:", round(maxvals[maxiter],4))
}

#par(mfrow = c(1,1))
#maxiter <- c(10,20,30)
#mutprob <- c(0.1,0.2,0.3)
#for(i in 1:2){
#  for(j in 1:2){
#    genalgfunc(maxiter[i],mutprob[j])
#  }
#}
set.seed(-3456)
genalgfunc( maxiter = 100, mutprob = 0.5)
set.seed(-3456)
genalgfunc( maxiter = 100, mutprob =0.9)
set.seed(-3456)
genalgfunc( maxiter = 10, mutprob =0.1)
physical <- read.csv2("physical.csv", sep = "," ,header = TRUE, stringsAsFactors = FALSE)

#2.1
physical$X <- as.numeric(physical$X)
physical$Y <- as.numeric(physical$Y)
physical$Z <- as.numeric(physical$Z)
```

```r
plot(physical$X, physical$Y, type = "l", ylim = c(0, 35), xlab = "X", ylab="value",main = "Time series p
lines(physical$X, physical$Z, col="red")
legend("topright", c("Y vs X", "Z vs X"),
       lty=c(1,1), lwd = c(2.5, 2.5), col = c("black", "red"))
em <- function(Y, Z, X){
Zobs <- Z[!is.na(Z)]
Zmiss <- Z[is.na(Z)]
Xobs <- which(!is.na(Z))
n <- length(Z)
r <- length(Zmiss)
# Initial value
lambda <- 100
i <- 1

repeat{
  # E- step
  EY <- sum(Y*X) / 2
  EZo <- sum(Zobs*X[Xobs]) / 4
  EZm <- (r * lambda) / 2
  # M - step
  lambda1 <- (EY + EZm + EZo) / n
  # Stop if converged
  if ( abs(lambda1 - lambda) < 0.001) break
  lambda <- lambda1
  i <- i + 1
}
res <- data.frame(Iterations = i, Lambda = lambda)
return(res)
}


em_exp <- em(physical$Y, physical$Z, physical$X)

em_exp
plot(physical$X, physical$Y, type = "l", ylim = c(0, 35), xlab = "X", ylab ="value",main=c("Time series
            "with the EM-alg. expected values"))
lines(physical$X, physical$Z, col="red")
lines(physical$X, em_exp$Lambda/physical$X, col="blue")
lines(physical$X, (2*em_exp$Lambda)/physical$X, col="green")
legend("topright", c("Y vs X", "Z vs X", "EY vs X", "EZ vs X"),
       lty=c(1,1), lwd = c(2.5, 2.5), col = c("black", "red", "blue", "green"))
## NA
```